CLAIMS

We claim:

1. A method of handling an exception comprising:

recognizing that an exception has occurred during the execution of a given method;

consulting a stack frame associated with said given method to determine an identity of an object required to be unlocked; and

removing said lock on said object.

2. The method of claim 1 further comprising:

determining a program counter address at which said exception has occurred;

determining, from a table, a synchronization depth associated with said program counter address; and

wherein said consulting said stack frame includes reading from a location associated with said synchronization depth.

3. An exception handler operable to:

recognize that an exception has occurred during the execution of a given method;

consult a stack frame associated with said given method to determine an identity of an object required to be unlocked; and

remove said lock on said object.

4. A computer readable medium containing computer-executable instructions which, when performed by a processor in a computer system, cause said computer system to:

recognize that an exception has occurred during the execution of a given method;

consult a stack frame associated with said given method to determine an identity of an object required to be unlocked; and

remove said lock on said object.

5. At a just in time compiler of programming language code, said programming language code including a plurality of instructions, a method of generating executable code comprising:

determining a synchronization depth for said each instruction;

associating said synchronization depth with a program counter address associated with said each instruction;

determining a continuous range of program counter addresses associated with an equivalent synchronization depth; and

storing an indication of said continuous range of program counter addresses in a table associated with said equivalent synchronization depth.

6. The method of claim 5 wherein said programming language code is Java.

7. The method of claim 5 wherein determining said continuous range of program counter addresses comprises, where a first synchronization depth determined for a first instruction differs from a second synchronization depth determined for a directly preceding instruction, storing said second synchronization depth in said table associated with a given range of program counter addresses, where said given range of program counter addresses includes said program counter address of said directly preceding instruction.

8. The method of claim 5, where said plurality of instructions may be arranged in a plurality of basic blocks of code, further comprising determining a synchronization depth at the beginning of each said basic block of code.

9. A just in time compiler of programming language code, said programming language code including a plurality of instructions, said compiler operable to:

determine a synchronization depth for said each instruction;

associate said synchronization depth with a program counter address associated with said each instruction;

determine a continuous range of program counter addresses associated with an equivalent synchronization depth; and

store an indication of said continuous range of program counter addresses in a table associated with said equivalent synchronization depth.

10. A computer readable medium containing computer-executable instructions which, when performed by a processor in a computer system, cause said computer system to:

determine a synchronization depth for said each instruction;

associate said synchronization depth with a program counter address associated with said each instruction;

determine a continuous range of program counter addresses associated with an equivalent synchronization depth; and

store an indication of said continuous range of program counter addresses in a table associated with said equivalent synchronization depth.